

Client and Server Communication

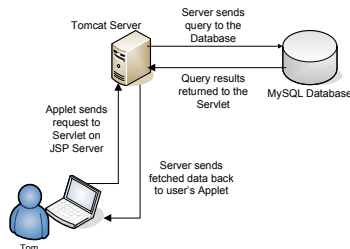
Charles Dairman

Problem/Questions

- There are many different technologies for web-based applications available.
- Which technology is best suited for the deployment of WITA?

Approach/Technologies

- Many different technologies were studied including: PHP, ASP, Applets, and JSP/Servlets.
- Weigh the advantages and disadvantages of each technology.
- Ensure the selected technology can support all of our requirements.



Outcome

- In the end, it was concluded that the combination of an Apache Tomcat JSP server with Applets executing on the Client's machine would prove to be the optimal solution for the server to client communication.

Client Interface

Seth Carpenter

Problem/Questions

- Provide current and historical weather information with traffic report obtained from government sources in a friendly graphical environment
- Create a client interface that is capable of displaying the dynamic components of the system (such as the traffic map and Doppler radar display)
- Support easy access from any computer

Approach/Technologies

- Create Java applets for each of the major components of the system (traffic, weather, and the combined quick view) to be viewed from a Tomcat JSP server
- Use JSP servlet to pass almanac and traffic information from the MySQL database to the applet
- Use multi-threaded applets to synthesize fast display of graphical data

Outcome

- JSP pages with embedded applets are viewable from any machine with the standard Java Virtual Machine
- Applets can connect to the database to retrieve and display relevant information
- Applets allow the user to interact with the dynamic data (various weather displays can be selected, the traffic map displays incident markers and will provide data for a selected route indicated by the user)

Data Processing

Robert Flasher

Problem/Questions

- Obtain near real-time weather and traffic information for the greater Phoenix area
- Format weather and traffic data for display by the client subprogram
- Maintain a weather almanac

Approach/Technologies

- Use object orientated software design fundamentals to develop a modular design where program control synchronizes data updates
- Use the Java computer programming language platform to implement the software design
- Obtain National Oceanic and Atmospheric Administration (NOAA) and Arizona Department of Transportation (ADOT) weather and traffic HTML web pages over the Internet
- Develop text parsing routines to extract data from the HTML text
- Develop graphics rendering routines to create graphics files with the extracted current weather and traffic information

Outcome/Outgoing Work

- Current weather condition, current traffic condition and Doppler radar graphics are updated every half hour and five day weather forecast, air quality, and Arizona ski report graphics are updated every morning
- High, low, and mean temperatures and the date are written to the almanac weather database at the end of every day
- Modular program implementation allows easy addition of modules to create new display types

Database

Felix Gorodishter

Problem/Questions

- Develop efficient database schema
- Develop efficient access/processing of data
- Analyze implementing/executing queries on remote servers with varying web technologies

Approach/Technologies

- Develop a simple database incorporating a readily available open-source database package – MySQL
- Based on this decision, most programming languages have developed classes/methods for interfacing with the database with synonymous syntax
- Becomes almost Language-Independent and Modular

Almanac Weather Database

CITY	MONTH	DAY	YEAR	MAX	MEAN	MIN	DEW	INDEX
PHOENIX	1	1	1973	53.6	44.9	37.4	0	1
PHOENIX	1	2	1979	57.2	45	37.4	0	2
PHOENIX	1	5	2005	53.6	47.9	39.2	0	5

Outcome/Ongoing Work

- Storage/retrieval code is easy to modify and transfer to different web based languages.
- Database is fast enough to support a multitude of users accessing the system in parallel.
- Similar database schema to be implemented for traffic mapping and coordinate → street name storage.